****

# Practices for Secure Software Report

Table of Contents

[Document Revision History 3](#_Toc102040754)

[Client 3](#_Toc102040755)

[Instructions 3](#_Toc102040756)

[Developer 4](#_Toc102040757)

[1. Algorithm Cipher 4](#_Toc102040758)

[2. Certificate Generation 4](#_Toc102040759)

[3. Deploy Cipher 4](#_Toc102040760)

[4. Secure Communications 4](#_Toc102040761)

[5. Secondary Testing 4](#_Toc102040762)

[6. Functional Testing 4](#_Toc102040763)

[7. Summary 4](#_Toc102040764)

[8. Industry Standard Best Practices 4](#_Toc102040765)

## Document Revision History

| **Version** | **Date** | **Author** | **Comments** |
| --- | --- | --- | --- |
| **1.0** | **2/25/24** | **Justin Osman** | **recommendations** |

## Client



## Instructions

Submit this completed practices for secure software report. Replace the bracketed text with the relevant information. You must document your process for writing secure communications and refactoring code that complies with software security testing protocols.

* Respond to the steps outlined below and include your findings.
* Respond using your own words. You may also choose to include images or supporting materials. If you include them, make certain to insert them in all the relevant locations in the document.
* Refer to the Project Two Guidelines and Rubric for more detailed instructions about each section of the template.

## Developer

Justin Osman

## Algorithm Cipher

I recommend SHA-256 to be used for the encryption algorithm cipher for Artemis Financial. SHA-256 is a widely used cryptographic hash function known for its strong security and reliability. Generating a 256-bit hash value, SHA-256 ensures data integrity by verifying the authenticity and integrity of transmitted files through checksum verification, which meets the criteria Artemis Financial laid out. Operating on blocks of 512 bits, SHA-256 employs logical operations like bitwise operations and modular addition to compute hash values deterministically. One thing that stands out with SHA-256 is its collision resistance, making it highly improbable for two different inputs to produce the same output hash. SHA-256 doesn't involve keys like symmetric or asymmetric encryption algorithms, and it doesn't require random numbers during computation.

SHA-256 was Initially published by NIST in 2001, SHA, it’s become a cornerstone of modern cryptographic protocols due to its strength against well-known attacks and widespread support across programming languages and cryptographic libraries. Integrating SHA-256 into Artemis Financials’ software application will enhance data transmission security and safeguard against unauthorized tampering or alterations during file transfers, improving the overall integrity and reliability of their operations.

## Certificate Generation

Insert a screenshot below of the CER file.

## A computer screen shot of a computer program Description automatically generated

## A screenshot of a computer Description automatically generatedA screenshot of a computer Description automatically generated

## Deploy Cipher

A screenshot of a computer

Description automatically generated

## Secondary Testing

Insert screenshots below of the refactored code executed without errors and the dependency-check report.

A computer screen shot of a program

Description automatically generated

A computer screen shot of a program

Description automatically generated

A computer screen shot of a program

Description automatically generated

A screen shot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

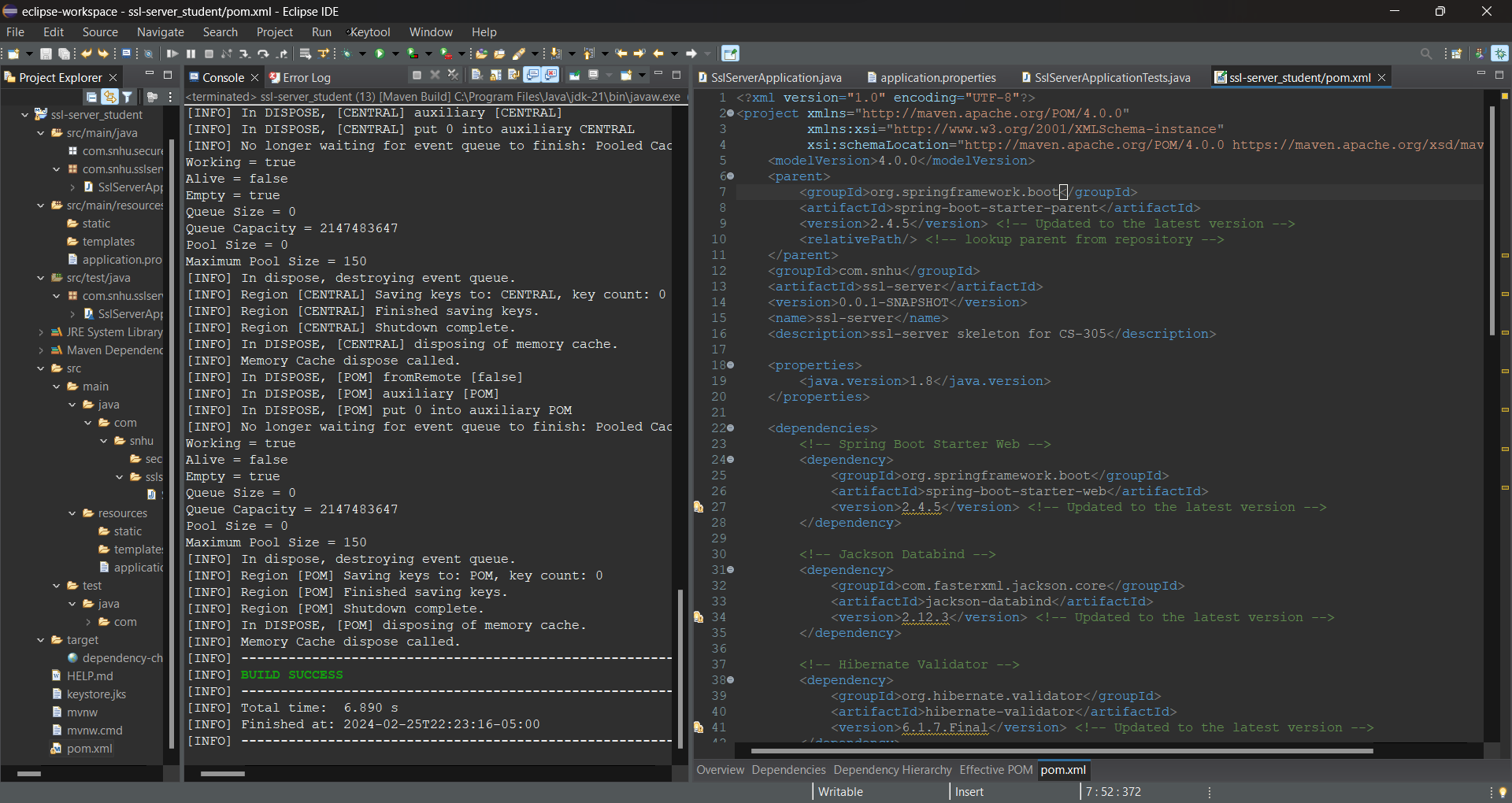
A screenshot of a computer

Description automatically generated

## Functional Testing

Insert a screenshot below of the refactored code executed without errors.

Updated pom.xml



A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

## Summary

Throughout the development lifecycle of the software application, we prioritized security by rigorously adhering to a Vulnerability Assessment Process Flow Diagram. This diagram served as a roadmap for identifying, analyzing, and mitigating security risks inherent to the application. By refactoring the code, we addressed several critical areas of security, notably input validation, authentication, and secure communication protocols. Through input validation, we ensured that all incoming data is properly sanitized, preventing SQL injection and cross-site scripting (XSS) vulnerabilities. Authentication mechanisms were strengthened to enforce strong password policies and secure user sessions, thus mitigating the risk of unauthorized access. Implementing secure communication protocols like HTTPS, alongside the configuration of SSL/TLS, safeguarded data in transit against interception and tampering.

The inclusion of the OWASP Dependency Check Maven plugin in our project's build configuration underscores our commitment to security. This tool automatically scans project dependencies for known vulnerabilities, allowing us to proactively address potential security flaws by updating or replacing vulnerable libraries. This process is a testament to our proactive approach in adding layers of security to the software application, ensuring that each component is scrutinized and secured against known vulnerabilities.

.

## Industry Standard Best Practices

Adhering to industry-standard best practices for secure coding has been pivotal in maintaining and enhancing the software application's security posture. By following guidelines and recommendations from reputable sources such as OWASP (Open Web Application Security Project) and CWE (Common Weakness Enumeration), we have established a strong framework for secure coding. This includes practices such as regular dependency checks, code reviews with a focus on security, and the integration of security testing within the CI/CD pipeline. These practices not only help in identifying vulnerabilities early in the development process but also instill a culture of security awareness among the development team.

The value of applying these best practices extends beyond the technical realm, contributing significantly to the company's overall wellbeing. Secure coding practices mitigate the risk of data breaches and cyber-attacks, which can lead to financial losses, legal repercussions, and damage to the company's reputation. By investing in security, the company demonstrates its commitment to protecting stakeholders' interests, including customers, employees, and partners. Furthermore, a strong security posture can be a competitive advantage, creating trust and confidence among existing and potential clients.

In conclusion, our approach to refactoring the software application, guided by a thorough vulnerability assessment process, and reinforced by industry-standard best practices for secure coding, represents a comprehensive strategy for safeguarding against known and emerging security threats. This commitment to security is not just about protecting data and systems but also about ensuring the long-term success and resilience of the company in the face of evolving cyber threats.